Group #6 Team Project Week#2 Journal

**Student Name**: Yuanyuan Jia **Student ID**: 010813322

**The work I did last week:**

**2016.4.11** Decide to use the cloud9 as IDE and install the MongoDB in the AWS EC2 instance. After research on MongoDB based on RESTful API programming, decide to use nodejs, expressjs and mongoose to implement the RESTful API. Set up the environment in Cloud9 and create a simple Node.js, express and Mongoose project to test.

**2016.4.12** Create models (user, order, comment etc.) based on the project data model designed last week. And implement 4 basic CRUD APIs for each model (GET, POST, PUT, DELETE by the primary id).

**2016.4.13** Perfect the catalog and dish model, use ref document to connect the two documents together and create other necessary APIs, such as get dishes by catalog, and get specific field from catalog.

**2016.4.14** Perfect the comment model, correct the primary id design for other models and create more necessary APIs for the front end to use comment model.

**2016.4.15** Perfect the user model, create embedded document between user and delivery address schema, so the data model is more clear and effective. Create more necessary APIs for the front end to use user model.

**2016.4.16** Discuss with team member Xing Yang, decide we need to add shopping cart as an additional model and correct the branch model.

**The problems I encountered:**

1. Because I plan to use Cloud9 as IDE to implement the database engine, but I cannot figure out what is the public DNS of cloud9, so I can test with PUT, POST, GET and DELETE. So I install the httpie on cloud9.
2. After I define the Mongo uri in the app.js file, it cannot connect to the MongoDB in the EC2, then I go to the EC2 and repair the Mongo server.
3. When it comes to the embedded document and the ref document, I spent some time to figure out how to add, delete, update sub document. Such as Mongoose populate return undefined, cannot update the sub document.
4. There are several modes need to record the create and update time, after research, I find that the [mongoose-createdat-updatedat](https://www.npmjs.com/package/mongoose-createdat-updatedat) plugin can automaticall add createdat and updateat in the schema. Also we need to compare these time stamp.

**The work I plan to do next week:**

1. Continue to work on the API implementation on the shopping cart, inventory, order and rest models and improve the models we have finished.
2. Will publish the finished APIs to the front end team.
3. Will insert more data into database.